J Supercomput (2017) 73:993–1016
DOI 10.1007/s11227-016-1731-x

Evolutionary Multiobjective Optimization algorithm for multimedia delivery in critical applications through Content-Aware Networks

Jordi Mongay Batalla[1,2] · **Constandinos X. Mavromoustakis**[3] · **George Mastorakis**[4] · **Daniel Négru**[5] · **Eugen Borcoci**[6]

Published online: 4 May 2016

Abstract Critical applications which need to deliver multimedia through the Internet, may achieve the required quality of service thanks to the Content-Aware Networks (CAN). The key element of CAN is an efficient decision algorithm responsible for the selection of the best content source and routing paths for content delivery. This paper proposes a two-phase decision algorithm, exploiting the Evolutionary Multiobjective Optimization (EMO) approach. It allows to consider valid information in different time scales, adapting decision-maker to the evolving network and server conditions as well as to get the optimal solution in different shapes of Pareto front. The simulation experiments performed in a large-scale network model, confirm the effectiveness of the proposed two-phase EMO algorithm, comparing to other multi-criteria decision algorithms used in CAN.

Keywords Content-Aware Networks · Critical applications · Evolutionary Multiobjective Optimization · Multi-path multi-source · Content server selection

✉ Jordi Mongay Batalla
jordim@interfree.it
http://www.nit.eu/z3

1 Seidor S.A., Barcelona, Spain

2 National Institute of Telecommunication, Warsaw, Poland

3 University of Nicosia, Nicosia, Cyprus

4 Technological Educational Institute of Crete, Heraklion, Greece

5 CNRS/LaBRI, University of Bordeaux, Talence, France

6 University Politehnica of Bucharest, Bucharest, Romania

1 Introduction

Critical applications (e.g., some surveillance and e-health applications) that need to download multimedia content from distant servers through the Internet have strong quality of service (QoS) requirements for the content transfer. The multimedia delivery should take into account the state of the network, server and end user in order to prepare the best delivery before starting the connection. This is possible thanks to the recently proposed Content-Aware Networks (CAN) [1], which are an evolutionary step forward in the content delivery through the Internet.

CAN architectures allow different actors involved in the content delivery (end users, service providers as CDN or conventional cloud, content providers, network provider) to arouse/sharpen the awareness concerning the whole delivery process. These features are crucial in critical applications as, e.g., surveillance. In surveillance applications the content is sent from the camera sensor to a content server through the Internet and from the server to the end user, which analyzes the images in real-time. With CAN architecture, the network becomes content aware which makes feasible the resource allocation and the reduction of end-to-end delay (for real-time viewing from the user). Thus, the application layer obtains network awareness rendering the adaptation more effective; i.e., the surveillance camera is able to arrange the content codification based on the state of the network in order to take the maximum profit from the network. Moreover, the service provider (e.g., the provider of surveillance images) receives the information about user profile and user context (i.e., information related to the terminal of the user viewing the surveillance images and the access network where she/he is connected) and this, in turn, makes the decisions about content transmissions more effective and fast adaptive.

Traditionally each actor performs only tasks directly related to her/him. For example, the network providers (ISP) were responsible for carrying the traffic without knowing the traffic requirements. On their part, the service providers usually built overlay infrastructures without any knowledge of the underlying network resources as, e.g., the access conditions of the user's terminal. This imposed serious limitations upon the content delivery performance in the Internet.

CAN architectures overcome this inefficiency establishing cooperation between the actors, by exploiting the distributed information systems designed on federation principles. The meeting point for all of them is an intermediate layer often called, in short, CAN. We can define CAN as an overlay, multi-domain network which may offer the framework for several actors' cooperation, where the service providers manage the high-level services, content providers manage the content servers, network providers manage connectivity resources at the underlying level, and end users consume the content [2]. This approach allows for a cross-layer management and optimization thus offering higher performance in content delivery. Business models managing these relations are flexible, e.g., the network provider may be also a service provider implementing the CAN in its own infrastructure and offering high-level services.

CAN demands the knowledge of the status of content servers and network conditions gathered from monitoring of the number of performance metrics. Moreover, metrics are correlated and this correlation is, in general, unknown. Therefore, new approaches dealing with multi-parameter environment should be considered in CANs.

Current algorithms for content source (also called server in this paper) selection are generally based on multi-criteria decision making (MCDM), which proposes quite a straightforward solution: the definition of mapping of the objective functions onto a single aggregated scalar function [3].

In this paper, we introduce Evolutionary Multiobjective Optimization (EMO) in order to properly select the content sources (containing multimedia content for critical applications) according to the characteristics of the selection process used in CAN. The presented EMO approach admits the selection of servers that are distributed along the Pareto front in opposition to MCDM approach, which only allows for the selection of content servers with similar characteristics (not optimal in CAN systems). The main objective of the EMO algorithm for content server selection in CAN is not to find the pure optimal Pareto front but the primary purpose is to find the solutions that are widespread on it.

The paper is structured as follows: after presenting the characteristics of CAN in Sect. 2, we present the definitions and theorems (Sect. 3.1) used for defining the EMO algorithm, which is presented in Sect. 3.2 (overall view), Sect. 3.3 (first phase of the algorithm) and Sect. 3.4 (second phase of the algorithm). The algorithm is validated via simulations in Sect. 4, along with a comparison of its performance with random selection and with domination-based selection algorithms. The paper is concluded in Sect. 5.

2 Related work and research motivation

CANs enable network-, content-, server- and user context-awareness, to achieve an enhanced architecture for multimedia content delivery. The information feeds the decision algorithms which choose the pertinent configuration such as the best source or the suitable delivery path. The input information relevance and the efficiency of the decision algorithms are crucial for the system performance.

One of the research challenges in CAN is the decision process selecting the best content source (from where the critical multimedia content will be downloaded). The decisions could be taken by the client applications, by the content provider (server side) or by the network infrastructure.

In approach from the side of client, there is an application which selects the best content based on the information collected by itself. The [4,5] exploit the dynamic probing and statistical estimation of different metrics such as round trip delay, available bandwidth, or servers' responsiveness. The results presented in [4] confirm that even simple dynamic probing outperforms blind client-side selection. However, the main drawbacks are the limited performance and weak scalability in an Internet-wide deployment.

The server side selection strategy [6] aggregates information at the server side and uses it to redirect the content requests coming from specific physical areas. The scalability requirements for server selection ask for some offline pre-computation as a preparation to serve content requests in real time.

One solution based on network-level decisions is "route-by-name" [7,8] approach, assuming that every CAN node forwards the content request towards the destination

source based on its local knowledge. The decision on source selection is taken in a distributed way as a concatenation of local optimizations. Therefore, the final solution may not be optimized in the global scope.

The DNS-like approaches [9–12], collect information on available content replicas (including codec information), content server status and network conditions, then use it for best content source selection, achieving in principle a globally optimal solution. However, an intermediate service layer is needed (called simply CAN) for managing the information that arrived from different entities and controlling the end-to-end content delivery.

In order to achieve the network-, content- and user-awareness, CAN should select, among all possible metrics (e.g., transfer delay, available bandwidth, packet loss, distribution mode, adaptation capabilities, load of the content server, bandwidth on access link, terminal context for user awareness), a subset that has the greatest impact on the quality of the given content type, and associate them with an acceptable cost model [13]. Jung et al. in [14], demonstrated that centralized decisions—taken on the basis of the knowledge about the delivered content and network conditions—optimize the utilization of the network resources and improve QoE on the user side.

Such an improvement requires an optimization decision algorithm considering possible solutions (e.g., a number of content servers, the different bitrates to download the content, etc.) and deciding about the best one for the current network conditions. The decision is, in general, a NP-complete problem, since it results in a multi-criteria decision problem which, normally, is not polynomially solvable. In practice, heuristics are usually used to compute a sub-optimal solution (a common one consists in the reduction of the multi-criteria vector to a scalar value by using an appropriate cost function, e.g., [15]).

The optimization decision algorithm can be centralized in the CAN Management and Control (CAN M&C). The latter takes mid-long- and short-term scale actions. Some processes are directed to access the content and prepare the network for the content delivery. The CAN M&C establishes (for days or weeks), the paths that will potentially connect end users and servers. These paths are normally specified in the SLS with the service provider and can consider QoS guarantees (strict, relative, best effort only). It is the responsibility of the CAN M&C to choose the best potential paths for the future content delivery. After paths setup in the network, the users can request specific content and, once again, the CAN M&C will decide about the best path/s and best source/s for serving the requested content to the user (short-term scale). Generally, the final decision of the content source is acknowledged to be the responsibility of the service provider or the content provider (another possibility is that the end user takes the final decision by using, e.g., HTTP-based protocols with Multi-Base URL requests). Therefore, the CAN decision algorithm should give out a limited (several) set of potential paths and sources which will be used by the service provider for optimizing the delivery on the basis of, e.g., the user profile.

The path selection optimization (besides source selection) is, currently, a hot issue in big content distribution networks. Some solutions are investigated to control the delivery path by the CDN infrastructure [16]. The end-to-end path is characterized by a concatenation of inter-domain paths, intra-domain paths being excluded.

A decision algorithm in CAN should consider the two aforementioned processes [17]. In the first phase, the algorithm should select a number of end-to-end paths (connecting the user and any server with the requested content) with their respective transfer characteristics. This is executed offline because of scalability limitations (it is not possible to discover paths on per-request basis!). Example parameters defining the paths are bandwidth of the path and delay dimensioned for the path. The second phase of the selection process is activated for each content request to find a limited number of content sources (and attached delivery paths) for serving the request. Both phases should be passed through to achieve the same objective.

This two-phase EMO algorithm is presented in the next section. In the first phase, a number of paths are selected for serving the high-popularity content requests in the future (i.e., when the content requests arrive). The second phase selects the actual content source. Example parameters defining the content source are the load of the server and its access bandwidth. Other authors ([18–20]) presented two-phase approaches for EMO but, in those cases, the two phases tried to increase efficiency of the algorithm itself. In our case, the two phases are mandatory by the same nature of CAN and the approach to solve the problem is completely different.

EMO popularity has increased also in the telecommunications field. Thus, mobile communications [21–23] and neural [24] and sensor [25] networks include research works which consider EMO as an efficient tool for dimensioning network infrastructures [26].

3 EMO algorithm for source selection in CAN

3.1 Definitions

The parameters used for monitoring the network and sources in CAN are the sum calculated or the minimum of different other values. For example, the delay of the path is the sum of the delays in all the links of the paths. The same occurs for jitter (approximation for very low values of jitter), losses, etc. Other values as the bandwidth in the path are calculated as the minimum of the bandwidth in all the links. The content source is characterized, among others, by server load which is a function of the sum of all the connections served by the source. Generally, the objective of the decision algorithm is to minimize the values of some parameters (delay, jitter, source load,…).

Therefore, without any loss of generality [27], we define the *ε-dominance* with the following additive expression.

Definition 1 (*ε-Dominance*) Let be $a, b \in X \subset \Re^n$. *Then* a is said to ε-dominate b for some $\varepsilon \subset \Re^n > 0$, denoted as $a \succ_\varepsilon b$, if

$$a_i - \varepsilon_i \leq b_i \quad \forall i \in \{1, \ldots, n\} \tag{1}$$

For describing the two-phase process selection that is used in CAN, we should assume that the first phase of the selection process is performed on the basis of n parameters, whereas the second selection is performed on the basis of m ($m > n$) parameters. So, we define the broad solution as follows:

Definition 2 (*Broad solution*) Let be $a \subset \Re^n$. *Then* $a^* \subset \Re^m$ $(m > n)$ is said a broad solution of a if

$$a_i^* = a_i \quad \forall i \in \{1, \ldots, n\} \tag{2}$$

Definition 3 (*Broad population*) Let be $X \subset \Re^n$. *Then* $X^* \subset \Re^m$ $(m > n)$ is said a broad population of X if

$$\forall a \in X, \quad \exists a^* \in X^* | a^* \quad \text{is a broad solution of } a \tag{3}$$

Definition 4 (*Probability of ε-dominance*) The probability that broad solution $a^* \in R^m$ ε-dominates ($\varepsilon \in R^m$) broad solution $b^* \in R^m$ denoted as $P(a^* \succ_\varepsilon b^*)$, is a value between 0 and 1 and corresponds to the probability of fulfilling the formula (4):

$$a_i - \varepsilon_i \leq b_i \quad \forall i \in \{1, \ldots, m\} \tag{4}$$

From these definitions we can derive the following (rather evident) theorems, which are crucial for the specification of the proposed algorithm.

Theorem 1 *If solution a does not ε-dominate solution b, then broad solution a^* does not ε^*-dominate broad solution b^*, where $\varepsilon_i^* = \varepsilon_i, i = \{1, \ldots, n\}$*

Proof If solution a does not ε-dominate solution b, then:

$$\exists i \in \{1, \ldots, n\} | a_i - \varepsilon_i > b_i \text{ and, since } a_i^* = b_i^* \ \forall i = \{1, \ldots, \mathrm{n}\}, \text{ then : } a_i^* - \varepsilon_i^* > b_i^*$$

and a^* does not ε^*-dominate b^*.

Theorem 2 *If solution a does ε-dominate solution b, then $P(a^* \succ_\varepsilon^* b^*)$ depends only on the parameters $n + 1, \ldots, m$, where $\varepsilon_i^* = \varepsilon_i, i = \{1, \ldots, n\}$*

Proof

$$\begin{aligned} P(a^* \succ_{\varepsilon^*} b^*) = {} & P(a^* \succ_{\varepsilon^*} b^* / a \succ_\varepsilon b) \times P(a \succ_\varepsilon b) \\ & + P(a^* \succ_{\varepsilon^*} b^* / a \not\succ_\varepsilon b) \times P(a \not\succ_\varepsilon b), \end{aligned}$$

since $P(a^* \succ_{\varepsilon^*} b^* / a \not\succ_\varepsilon b) = 0$ (from Theorem 1) and a ε-dominates b, i.e., $a_i - \varepsilon_i \leq b_i$, $\forall i \in \{1, \ldots, n\}$, then

$$\begin{aligned} P(a^* \succ_{\varepsilon^*} b^*) &= P(a_i^* - \varepsilon_i^* \leq b_i^*, \forall i = n+1, \ldots, m) \times P(a \succ_\varepsilon b^*) \\ &= P(a_i^* - \varepsilon_i^* \leq b_i^*, \forall i = n+1, \ldots, m) \end{aligned}$$

Corollary *In the case when the parameters $n + 1, \ldots, m$ are independent of the parameters $1, 2, \ldots, n$, then it is not possible to conclude ε^*-dominance by only knowing ε-dominance.*

In the case of CAN, the first n parameters could be (for example) bandwidth and delay of the path (for path selection), whereas the m parameters of the broad population would be bandwidth and delay of the path and, on the other hand, load and access bandwidth of the server (for server selection).

3.2 Overall definition of the algorithm

We propose a two-phase evolutionary multiobjective optimization algorithm for selecting the content sources in CAN infrastructure. The first selection is performed offline and selects the paths and the distribution modes (related to streaming protocols) attached to each path on the basis of static information of the network and server access to the network. The second selection is performed when an end user request arrives to the M&C of CAN. The selection manager initiates the second phase of the content source selection algorithm by using the dynamic information from the network and servers. This information uses real-time measurements of the state of the servers (and, eventually, users) since network information is much more sensitive and, normally, is not disclosed by the network operator. As a result, the second phase of the algorithm offers a number of optimized content sources which can serve the user request.

After this two-phase selection performed by CAN, the service provider will choose the final source to serve the request. This selection will be based on user profile and terminal capabilities, which will be known only in a posterior stage by, e.g., authorization requirements. As an example, consider a CAN that has a Service Level Agreement with a third-part service provider, which is the final operator offering the service to the end user. It is beyond the scope of this paper to cope with the last selection process performed by the service provider. The proposed EMO algorithm deals with the first two phases.

There are three main reasons for providing an EMO algorithm to solve source selection in CAN: (1) the Pareto front may present different shapes, specifically it may be concave or even disconnected, which is not easily solved by traditional mathematical programming techniques [28], (2) EMO finds more than one solution. This is the case of CAN M&C, which should leave the last decision to an external actor (service provider or end user) which selects the final source and (3) the final solutions can be spread along the Pareto front, which is not achievable by MCDM algorithms currently used in CAN. Let us remark that the scope of EMO algorithm for content source selection is not to find the theoretical optimal Pareto front but to find several sources that are quasi-optimal and are distributed along the whole Pareto front.

The EMO algorithm proposed in this paper takes some ideas of the different, well-known, EMO algorithms and adapts them for CAN content source selection problem. From Pareto Archived Evolution Strategy [29], we inherit the placement of solutions in a certain grid location and the use of this grid for classifying the fitness of the solutions. Moreover, for variation of the population, we use an approach similar to the continuously updated fitness sharing [30] used in Niched Pareto Genetic Algorithm 2 [31]. From Strength Pareto Evolutionary Algorithm 2 (SPEA2) [32,33], we take the idea of fine-grained fitness assignment policy which considers the number of dominated and dominating solutions. Moreover, we adopt the strategy of calculating the distance with the closest neighbors in order to increase the efficiency of the algorithm.

By the term "efficiency", we understand the capability of the algorithm for finding solutions distributed along the Pareto front.

We introduce some novel mechanisms that are specific for the CAN problem that we tackle in this paper. The first phase for the selection of paths introduces a new

fitness assignment in accordance with the content distribution in the network. The second phase includes an elitism mechanism for assuring that the final solutions are not far from the solutions presented by the CAN system and a mechanism for setting the rank of some parameters that could be dynamically adapted by the CAN M&C.

Several new EMO algorithms and many modifications have been presented during the last years. These novel approaches overcome the results of the classical ones but, they search higher efficiency and/or simplicity in finding the Pareto front (e.g., [34,35]), which is not the scope of content source selection algorithm.

The proposed algorithm is a two-phase algorithm, where each phase corresponds to each selection in CAN, i.e., path selection and content source selection. The algorithm aims at finding the solutions of the Pareto front which are not far from the content sources proposed by CAN M&C and, at the same time, which are shared along the Pareto front in order to provide enough variety in the range of possible output solutions.

In the first phase, CAN establishes the potential paths between sources' domains and users' domains. Generally, users and servers may be within the same domain (e.g., network operators which have their own CDN infrastructure) and, in this case, the paths are intra-domain. In order to decide which paths will be configured, the CAN M&C obtains information about the links (inter-domain) and the service level agreements with different network Operators. Such information is taken in a long-term scale (static information) and is grouped in a number (n) of parameters (provisioned delay, provisioned loss rate, number of hops, cost, etc.) which are the basis of the path selection process.

Once the paths are selected and configured (path establishment), the users can generate requests of content that will be served by using the established paths. The selection of paths limits the final selection of content source since only the servers whose domain is connected to the users by one of the selected paths can be chosen during the second phase of the algorithm.

The second phase is the selection of content source/s which may potentially stream the requested content when a user sends the request. This second phase is performed per content request and uses real-time (dynamic) information in addition to static one. Dynamic information characterizes the state of the paths (current delay, current packet loss rate, etc.) and sources (current load, outgoing bandwidth of the server, etc.). For each path established after the first phase, we can find several content sources that potentially may serve the content by using the same path. In other words, there could exist several content sources with the same values of the n first parameters and different $(m - n)$ last parameters.

The output of the second phase is a limited (K_2) number of potential sources that contain the requested content and fulfill the requirements of the connection (e.g., live streaming). These potential sources are optimized within the Pareto set.

3.3 First phase (path selection)

The first phase of the algorithm runs along with the CAN establishment. The system receives information about the dimensioning parameters of the paths and selects a

number K_1 of paths between the users' domain and servers' domain containing high-popularity content.

The initial population of solutions in the first phase is a number of potential end-to-end paths connecting the users' domain and the servers' domains containing the high-popularity content. Each path is defined by n parameters.

3.3.1 Requirements for the first phase selection process

The objectives of the path selection are two: (1) the algorithm should choose the best paths which will connect the best sources for serving the content requests; (2) the algorithm should allow for the selection of different sources situated at different domains, so the algorithm should preserve the paths that connect the users' domains with different servers' domains.

Selection of unconnected paths Two paths connecting the same users' and servers' domains are called connected paths, whereas two paths connecting the same users' domain but two different servers' domains are called unconnected paths. An important characteristic of CAN is that two connected paths have rather similar values of several parameters (number of hops, delay, etc.), unlike unconnected paths.

We have checked this heuristic by simulations in a CAN simulation model that we presented in [17]. The model groups together content server and user population distribution into the Internet topology and assigns to the inter-domain links and to the servers some parameters (e.g., link bandwidth and server capacity) that are in accordance with current Video on Demand (VoD) streaming in the Internet.

This extensive model takes the parameters from the largest content and service providers. Specifically, the model assumes Internet topology and distribution of user population on the basis of University of Oregon studies [36] and RIPE data [37], which consider Tier 1, Tier 2 and Tier 3 domains; while the source distribution in the domains and the source characteristics (maximum number of streams) are modeled according to the Akamai infrastructure [38] and other important content providers, whose data (e.g., domain identifiers) are public. The model considers a network topology of around 36,000 domains and 103,000 links, whereas the source population counts more than 400,000 servers. We assigned the values of two dimensioning (static) parameters to each link: bandwidth (BW) and dimensioning (i.e., maximum) IP Packet Transfer Delay (IPTD). These values take in consideration the type of domains (Tier 1, 2 or 3) which the links connect, and are in accordance to the guidelines of [39,40].

The simulation methodology is based on the analysis of the content delivery process from the point of view of one exemplary Tier 3 users' domain. We find the shortest paths from that domain to all the other domains of the network topology. Note that between the users' domain and a given domain, several shortest paths with the same number of hops can be created. The value of the BW of each path (BW_{path}) is the minimum value of the BW of all the links belonging to the path, whereas the value of IPTD of each path ($IPTD_{path}$) is the sum of the IPTD values of all the links in the path (only inter-domain links are considered).

When the paths are found, we evaluate the parameters of pairs of paths. We select randomly 100,000 pairs of paths, all of them starting at the exemplary end users'

Table 1 Euclidean distance of parameters of randomly selected pairs of paths

Pair of paths	$d^2(a,b)_{\text{mean}}$
Connected paths	0.161 ± 0.112
Unconnected paths	0.761 ± 0.157

domain. For each pair of paths we first check whether these two paths end at the same sources' domain (connected paths) or at different sources' domains (unconnected paths). Let us remark that connected paths cross a higher number of common links than unconnected paths.

For each pair of paths, a and b, we evaluate the distance between them $d^2(a,b)$ by calculating the squared Euclidean distance of the two parameters: BW and IPTD as shown in formula (5).

$$d^2(a,b) = \left(\frac{\text{BW}_a - \text{BW}_b}{\text{BW}_{\max} - \text{BW}_{\min}}\right)^2 + \left(\frac{\text{IPTD}_a - \text{IPTD}_b}{\text{IPTD}_{\max} - \text{IPTD}_{\min}}\right)^2, \tag{5}$$

where $\text{BW}_{\max}$, $\text{BW}_{\min}$, $\text{IPTD}_{\max}$, $\text{IPTD}_{\min}$ are the maximum and minimum values of BW and IPTD for all the paths.

We repeated the tests 25 times in order to obtain the confidence intervals of the mean values.

The mean values of $d^2(a,b)$ together with the confidence intervals at the 95 % confidence level for connected and unconnected paths are shown in Table 1. As we can observe, connected paths have significantly lower $d^2(a,b)_{\text{mean}}$, so the values of the parameters are much more similar than in the case of two unconnected paths.

The path selection algorithm should provide a mechanism to select the higher number of unconnected paths in order to offer different servers' domains (with different content sources) during the second phase of the selection algorithm. Therefore, the mechanism should select the solutions that are distant in the set population (unconnected paths). This is achieved in *Step 1*, as described below.

Selection of connected paths Between all the connected paths, only the best ones should be selected. The reason is the Thesis 1: for connected paths, domination in phase 1 implies domination in phase 2.

Proof If a and b are two connected paths, then they connect to the users the same content sources in the broad set population.

Let be $a_i^*, b_i^*, i = n+1, \ldots, m$ the set of parameters characterizing the content sources. If a and b are connected paths, then $a_i^* = b_i^*, i = n+1, \ldots, m$ (same parameters of content sources).

From Theorem 2, if $a \succ_\varepsilon b$, then $P(a^* \succ_{\varepsilon^*} b^*) = P(a_i^* - \varepsilon_i^* \leq b_i^*,\ \forall i = n+1, \ldots, m)$, since $a_i^* = b_i^*, i = n+1, \ldots, m$, then for any $\varepsilon_i^* > 0, a_i^* - \varepsilon_i^* < b_i^*$, quod erat demonstrandum.

Because of this, in the case of connected paths (characterized by proximity in the set population), the algorithm should exclude dominated solutions in the first phase of the selection process (path selection) and choose only the solutions that dominate a

highest number of other solutions, as performed in, e.g., SPEA [41]. This is performed in the *Step 2* of the first phase of the algorithm.

3.3.2 Features of the first phase algorithm

The algorithm starts with locating the set population in a n-dimensional grid as shown in Fig. 1 ($n = 2$). Each solution is located in the n-dimensional grid based on the values of the parameters in a similar way as a geographical location. The solutions included in different equilateral hyper-surface of n dimensions (each square in Fig. 1) are considered as unconnected paths and at least one solution of each hyper-surface (if any) will be selected in the first phase of the algorithm. Within one hyper-surface, all the paths are considered as connected and the dominating solutions are preferred.

So, the algorithm selects at least one solution of each hyper-surface, if any, and non-dominated solutions inside of the same hyper-surface, if needed (to complete the number of solutions).

3.3.3 First phase algorithm

In this section, we present the steps of the first phase algorithm that works as explained above.

Let K be the number of solutions in the population set and K_1 the number of solutions which will be selected in the first phase (archive size).

Let x_{ij} be the value of the parameter i (e.g., number of hops), $i = 1, \ldots, n$ of the solution (path) j, $j = 1, \ldots, K$.

Step 1: Initialization. Generate an initial population P_0 with all the potential paths. Generate an empty archive A. Calculate α_i as shown in (6), which determines the number of hyper-surfaces in each axe of the grid (see Fig. 1).

$$\alpha_i = \frac{\max_j x_{ij} - \min_j x_{ij}}{\text{int}\{\sqrt[n]{K_1}\}}; \quad i = 1, \ldots, n; \quad j = 1, \ldots, K \tag{6}$$

Reorder P_0 by calculating the value y_{ij} (normalized population) (see Fig. 1) classifying the solutions in the grid.

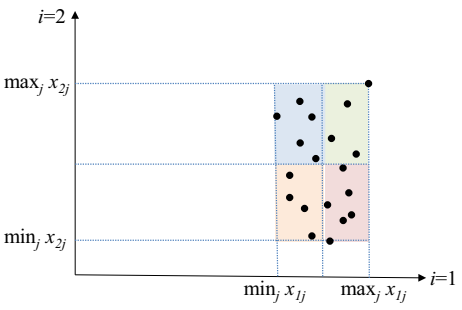

Fig. 1 Grid of the decision space (exemplary values: $n = 2$; $K = 19$; $K_1 = 8$; $\text{int}\sqrt{8} = 2$, as defined in the next sub-section)

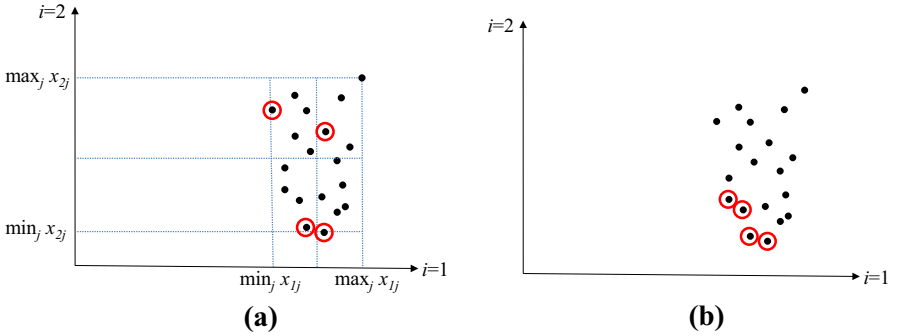

Fig. 2 Exemplary solutions ($n = 2$, $K_1 = 4$) selected by **a** first phase of the algorithm, **b** dominance-based algorithm

$$y_{ij} = \begin{cases} \text{int}\left\{\frac{x_{ij} - \min_j x_{ij}}{\alpha_i}\right\} + 1; & \text{if } x_{ij} \neq \max\limits_j x_{ij} \\ \text{int}\left\{\frac{x_{ij} - \min_j x_{ij}}{\alpha_i}\right\}; & \text{if } x_{ij} = \max\limits_j x_{ij} \end{cases} \quad i = 1, \ldots, n; \quad j = 1, \ldots, K \tag{7}$$

Create support vector y with n elements. Initialize $y = (1, 1, \ldots, 1)$.

Step 2: for all the solutions with $y_{ij} = y_i$:/* solutions in the same hyper-surface (square in Fig. 1)

- calculate the fitness value f_j as the number of solutions that are ε-dominated by the solution j, divided by the population size plus one.
- copy the solution x_{ij} (with $y_{ij} = y$) which has the highest value f_j to the archive. In the case when two or more solutions have the same highest value f_j, then copy one of them (random). Cut the solution x_{ij} from the population set.

Step 3: if $y = (\max_j y_{1j}, \max_j y_{2j}, \ldots, \max_j y_{nj})$ then $y = (1, 1, \ldots, 1)$ /* if all the hyper-surfaces were finished and not all the K_1 solutions have been selected, then the algorithm repeats the process from the first hyper-surface.

Increment index i in y_i.

Step 4: if size$(A) = K_1$, then end; else go to Step 2.

Figure 2 shows the differences between the presented first phase of the algorithm and the classical dominance-based algorithms. Figure 2a presents the exemplary solutions chosen by the first phase (encircled points), whereas Fig. 2b shows the (exemplary) solutions selected by an algorithm that prefers dominant solutions (encircled points).

3.4 Second phase

The second phase of the algorithm runs when a user requests any content to the system. The system receives information about the servers that store the requested content and selects a number K_2 of sources for serving the request. Generally, K_2 is a small value that allows for a final selection of content source by an external entity as, e.g., service provider. In some cases, K_2 may be equal to 1 (CAN decides the final content source).

The initial population of solutions in the second phase is a number of sources containing the requested content and connected to the users' domain by one of the K_1 paths which were the output of the first phase of the algorithm. According to the definition 2, the initial population of the second phase is a broad population of the output of the first phase. The solutions of the second phase have m ($m > n$) parameters.

3.4.1 Requirements for the second phase selection process

The proposed algorithm should introduce mechanisms that are specific for content source selection in CAN. On the one hand, CAN M&C should be able to control the rank of the parameters in the decision process and, on the other hand, elitism mechanisms should be introduced for preserving solutions that are close to the initial population, i.e., the results of the first phase.

Rank of parameters Since the state of the sources and the links within the CAN infrastructure is very dynamic, CAN M&C introduces monitoring elements for the real-time control of the state of the system and takes high-level decisions in order to introduce stability (for example, content provisioning of high-popularity content), when required. Also the selection of the content source should be part of such decisions since the CAN M&C may decide, for example, to solve network overload situations or to introduce source load balancing. Because of this, the second phase algorithm should be able to select the server taking into account the high-level decisions of the CAN M&C.

In our algorithm, we introduce a mechanism that allows some parameters to gain importance in the decision process, i.e., some parameters have higher rank within the algorithm. In this way, the CAN M&C may optimize the algorithm for the parameters of the top rank. A change of the weights ($w_i, i = 1, \ldots, m$) results in a change of the incidence of the content source selection into the system state. For example, the CAN M&C could introduce a higher weight of the source load in order to ensure server load balancing. This is achieved in the *Step 5* of the second phase algorithm presented below.

Elitism mechanism The Pareto optimal front discovered by the algorithm should be not far from the solutions of the initial population P'_0, which are the content sources that might serve the user request. Therefore, we introduce an elitism mechanism for preserving solutions which are near to the initial population and, at the same time, discarding other solutions even if they are closer to the Pareto optimal front (proximity to the Pareto optimal front is not the scope of the algorithm).

This elitism mechanism is based on gridding the decision space (as performed in the first phase) and marking the hyper-surfaces that contain at least one solution of the initial population. Solely the solutions belonging to the marked hyper-surfaces will be accepted in future populations. In this way, the algorithm sets bounds to the distance of the Pareto front found by the algorithm and the potential solutions offered by the content sources. Moreover, the elitism mechanism speeds up the convergence to the final solutions since it avoids further iterations of the algorithm. This is performed in *Step 7* of the second phase algorithm.

3.4.2 Features of the second phase algorithm

The second phase algorithm considers the paths selected in the first phase and extends this population with new parameters of servers' and users' domains. Among this broad population, a number of servers are selected in such a way that the servers are distributed along the Pareto front.

The evolutionary algorithm proposed for the second phase uses CAN-specific variation mechanism and environmental selection. The variation mechanism ensures that the children solutions are, at least, as good (solutions) as the parents ones. Any modification in the child (with regard to the parent) improves the parent since the algorithm assigns minimum values of one or more parameters (note that the multiobjective optimization consists of minimizing the values of the parameters). The proposed environmental selection assures that the new population is not far away from the original population P'_0.

The fitness assignment step applies dominance rank and count. In this way, the algorithm prefers solutions that dominate a higher number of solutions and, particularly, the algorithm discards dominated solutions. Note that in the case when the first phase of the algorithm selected dominated solutions and such dominated solutions were also dominated in the spread population of the second phase, then the fitness assignment of the second phase would eliminate such solutions when assigning to A'_1 in the first iteration. In other words, even when during the first phase, some dominated solutions could be chosen, afterwards such solutions are discarded if they are also dominated in the second phase.

3.4.3 Second phase algorithm

The steps of the second phase algorithm are the following:

Step 1: Initialization. Generate an initial population P'_0 (broad population of the output of the first phase). The single quotation mark of P'_0 is used for distinguishing from the initial population of the first phase P_0. Generate an empty archive $A'_0 = 0$. Generate a vector of weights of the m parameters w'_i, $i = 1, \ldots, m$. Calculate α'_i as shown in (8).

$$\alpha'_i = \frac{\max_j x_{ij} - \min_j x_{ij}}{\text{int}\{\sqrt[m]{K_2}\}}; \quad i = 1, \ldots, m; \;\; j \in P'_0 \tag{8}$$

Reorder P'_0 by calculating the value y'_{ij} as indicated in formula (7), which classifies the solutions in the grid.

Step 2: Calculate the fitness assignment F_i of each solution $i \in P'_t + A'_t$ (t is an iteration index with initial value equal to 0), where + stands for union of both the solution sets. F_i is calculated in the same way as SPEA2 [33], see formula (9).

$$F_i = \sum_{j \in P'_t + A'_t, j \succ_\varepsilon i} S_j + \frac{1}{d^2(i,l) + 2}, \quad i \in P'_t + A'_t \tag{9}$$

where S_j is the cardinality of the set: $\{j | j \in P'_t + A'_t \wedge i \succ_\varepsilon j\}$ and $d^2(i,l)$ is the squared distance from solution i to the next closest solution:

$$\min_{l \in P'_t + A'_t} \left[\sum_{k=1}^{m} (x_{ki} - x_{kl})^2 \right]$$

Step 3: Copy the K_2 solutions with lower F_i into A'_{t+1}.

Step 4: Matting selection: extract randomly two elements of P'_t. Copy the one with lower F_i into the matting pool (if F_i values of both elements are the same, then select one of them randomly) and cancel both of them from P'_t. Repeat the process until P'_t is empty.

Step 5: Variation: choose randomly two elements i and j of the matting pool and modify the solution i' in the following way: for each parameter $k, k = 1, \ldots, m$ select the minimum value between x_{ki} and $(x_{ki} + x_{kj})/2$ with a probability $w'_k/(w'_1 + w'_2 + \cdots + w'_m)$ (w'_i has been defined in *Step 1*) or leave the value x_{ki} with a probability $(1 - w'_k)$, as indicated in (10):

$$x_{ki'} = \begin{cases} \min\left(x_{ki}, \frac{x_{ki}+x_{kj}}{2}\right) & \text{with prob} = w'_k \\ x_{ki} \quad \text{with prob} = (1 - w'_k) \end{cases} \tag{10}$$

Copy the solution i' to the environmental pool.

Repeat the *Step 5* exchanging the values of the elements i and j/*in this occasion, the element j will be modified.

Cut solutions i and j from the matting pool.

Step 6: Repeat *Step 5* until matting pool is empty.

Step 7: Environmental selection: for each solution j' of the environmental pool, calculate $y_{ij'}$ as follows:

$$y_{ij'} = \begin{cases} \text{int}\left\{\frac{x_{ij'} - \min_j x_{ij}}{\alpha'_i}\right\} + 1; & \text{if } x_{ij'} \neq \max_j x_{ij} \\ \text{int}\left\{\frac{x_{ij'} - \min_j x_{ij}}{\alpha'_i}\right\}; & \text{if } x_{ij'} = \max_j x_{ij} \end{cases} \quad i = 1, \ldots, m; \quad j \in P'_0 \tag{11}$$

/* This action reorders the environmental pool population into the original grid of the second phase population P'_0.

Copy the solution j' to P'_{t+1} if and only if:

$$\forall i = 1, \ldots, m \quad \exists \text{ solution } k \in P'_0 | \quad y_{ik} = y_{ij'} \tag{12}$$

/* The algorithm only copies solutions that are in the same grid square as, at least, one original solution of P'_0.

Step 8: Termination condition: if the cardinality of the set P'_{t+1} is equal to 0 (P'_{t+1} is empty) or $t = T$ then go to *Step 9*, otherwise $t{+}{+}$, go to *Step 2*.

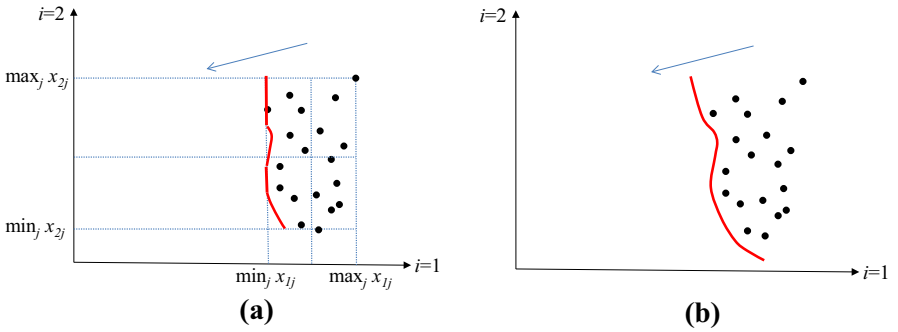

Fig. 3 Exemplary Pareto front found by **a** second phase of the algorithm; **b** classical EMO ($n = 2$)

Step 9: For each solution of A'_t, find the closest solution of P'_0. The furthers are the selected solutions for serving the content requests.

Figure 3a shows the Pareto front "found" by the presented second phase of the algorithm (red color), whereas Fig. 3b shows the Pareto front "found" by the classical EMO algorithm (red color). In the second phase of the algorithm the unique valid solutions are located inside the grid.

4 Performance evaluation of the algorithm by simulations

In order to validate the proposed algorithm, we provide two different analysis via simulations.

On the one hand, in Sect. 4.1 we argue that the novelties (compared to other EMO algorithms) which we have introduced in the second phase of the algorithm (variety, fitness and elitism) improve the content server selection process. Concretely, we verify when successive iterations of the algorithm provide solutions that are closer to the Pareto front as well as we investigate when *variety mechanism* allows for finding solutions which are optimal for selected parameter.

On the other hand, Sect. 4.2 compares the proposed two-phase EMO-based algorithm with MCDM-based algorithms for content server selection.

The simulation studies are based on the model of CAN which we have already used in the previous section. Once again, we consider the point of view of single users' domain (domain A). The algorithm optimizes path selection (first phase of the algorithm) by minimizing the inverse of $\mathrm{BW}_{\mathrm{path}}$ ($\mathrm{BW}^{-1}_{\mathrm{path}}$) and by minimizing the $\mathrm{IPTD}_{\mathrm{path}}$. For the second phase algorithm (content source selection) we consider two more parameters characterizing the servers: the server load, which is a random value $U[0.0, 1.0]$ calculated for each content server within the servers' domains and the inverse of the servers' domain access bandwidth ($\mathrm{BW}^{-1}_{\mathrm{access}}$). $\mathrm{BW}_{\mathrm{access}}$ is the BW of the link (belonging to the path) attached to the servers' domain. All the servers at the same servers' domain have the same value of $\mathrm{BW}_{\mathrm{access}}$.

The best content sources are the ones that have lower values of different parameters, i.e., near to $0 \in R^4$ ($\mathrm{BW}^{-1}_{\mathrm{path}} = 0$, $\mathrm{IPTD}_{\mathrm{path}} = 0$, load $= 0$, $\mathrm{BW}^{-1}_{\mathrm{access}} = 0$).

4.1 Analysis of second phase algorithm

The first tests show the selection of solutions in successive iterations of the second phase algorithm. Both simulation methodology and tool have been specified in Sect. 3.3.1. The tests assume that a number of $K_1 = 16$ paths have been selected. Each path communicates domain A with one servers' domain, where the servers are part of the initial population P_0'. Each path is characterized by the couple ($\mathrm{BW}_{\mathrm{path}}^{-1}$, $\mathrm{IPTD}_{\mathrm{path}}$), whereas each server is characterized by the quad term ($\mathrm{BW}_{\mathrm{path}}^{-1}$, $\mathrm{IPTD}_{\mathrm{path}}$, load, $\mathrm{BW}_{\mathrm{access}}^{-1}$). The value ε_i equals $(\max_j x_{ij} - \min_j x_{ij})/10$, $j \in P_0'$. The network scenario (which includes network topology and content source distribution) is only exemplary, so the conclusions are very limited and we cannot generalize.

Figure 4 shows how the solutions tend to the Pareto optimal front for successive iterations when all the weights of the parameters are equal to 0.5, i.e., all the parameters have the same probability of varying in successive iterations. In order to clearly show the results of the second phase algorithm, we calculated the weighted sum of the path parameters: $\mathrm{BW}_{\mathrm{path}}^{-1}/\max_{P0'}(\mathrm{BW}_{\mathrm{path}}^{-1}) + \mathrm{IPTD}_{\mathrm{path}}/\max_{P0'}(\mathrm{IPTD}_{\mathrm{path}})$. This sum is presented in X-axis of Fig. 4. The values $\max_{P0'}(\mathrm{BW}_{\mathrm{path}}^{-1})$ and $\max_{P0'}(\mathrm{IPTD}_{\mathrm{path}})$ are the maximum of the inverse of the path bandwidth and the path IPTD of the population P_0', respectively. Y-axis shows the values of the server parameters, i.e., $\mathrm{load}/\max_{P0'}(\mathrm{load}) + \mathrm{BW}_{\mathrm{access}}^{-1}/\max_{P0'}(\mathrm{BW}_{\mathrm{access}}^{-1})$, where $\max_{P0'}(\mathrm{load})$ and $\max_{P0'}(\mathrm{BW}_{\mathrm{access}}^{-1})$ are the maximum values of server load and the inverse of the access bandwidth of the population P_0', respectively. Even if the weighted sum representation could mislead the lector, the figures are useful to take some conclusions about the behavior of the algorithm.

Figure 4 presents the original solutions of $P_0'$and the $K_2 = 9$ solutions of successive archives ($A_1' - A_3'$). We can see that the solutions of successive archives are closer to 0, i.e., closer to the optimal solution. The selected solutions are the solutions belonging to P_0' that are closer to the solutions of A_3'. Note that in some cases the final solution (red circle) is different from the solution of A_1' (blue star) because of the functioning of *variety mechanism* (evolutionary), which finds optimal solutions. The surface marked

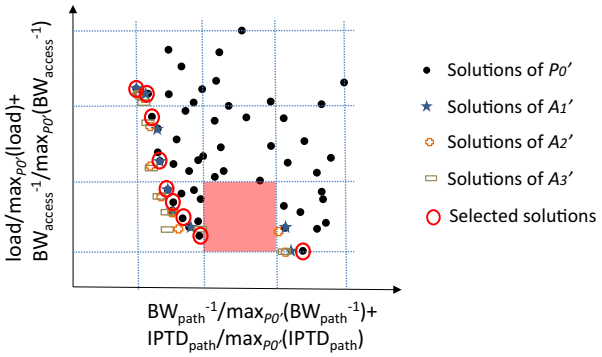

Fig. 4 Results of successive iterations of the second phase algorithm (archive solutions) for $w_1 = w_2 = w_3 = w_4 = 0.5$

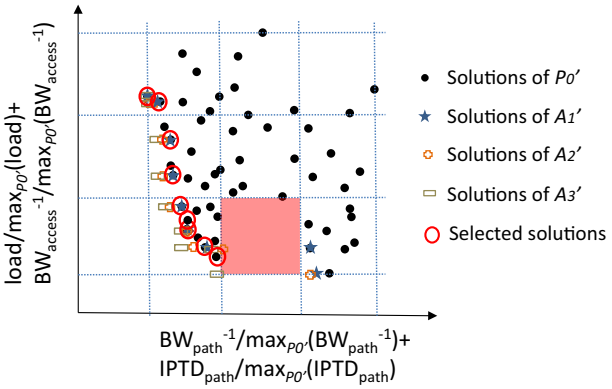

Fig. 5 Results of successive iterations of the second phase algorithm (archive solutions) for $w_1 = w_2 = 1$ and $w_3 = w_4 = 0$

with red color does not contain any solution of the population P_0'; therefore, any solution in A_i' is contained in this hyper-surface, which demonstrates that the elitism mechanism works properly.

The following test aims at demonstrating whether the CAN-specific *variety mechanism* works properly. We repeat the previous test (the same network scenario) with values of the weights are $w_1 = w_2 = 1$, $w_3 = w_4 = 0$. These weights should only provide a variety of the first two parameters: $\text{BW}_{\text{path}}^{-1}$ and $\text{IPTD}_{\text{path}}$, which can be observed in Fig. 5. The solutions of A_1' in Fig. 5 are the same as in Fig. 4 but, for successive iterations, the solutions are varying uniquely in the horizontal direction, which agrees with the values of the weights that we set in the algorithm. In fact, the values $w_3 = w_4 = 0$ do not allow for varying the solutions in the *Y*-dimension. We can conclude that the *variety mechanism* works as expected and provides to a different subset of selected solutions, which is optimal for $\text{BW}_{\text{path}}^{-1}$ and $\text{IPTD}_{\text{path}}$ (e.g., there is no selected solution on the bottom red hyper-surface of Fig. 1).

4.2 Comparison with other MCDM-based content source selection

Maximization of the extent of the obtained non-dominated front is one of the requirements for many EMO algorithms [42]. In the case of CAN, this requirement acquires special significance since CAN should supply a number of content sources with different characteristics to the service provider (or, eventually, end user) which is in charge of taking the final decision about the content source. This is, in fact, one of the main reasons for choosing EMO approach instead of MCDM approach for content source selection in CANs.

The following tests evaluate the efficiency of the proposed algorithm in the selection of solutions which are spread along the Pareto front. We compare the efficiency of the presented algorithm (with all the weights set to 0.5), called Domination-EMO or, briefly, D-E, with other three MCDM algorithms usually used in CAN infrastructures. Note that the scenario and the distribution of servers and users are the same in all

the tests and the only modifiable thing from one test to another is the implemented algorithm.

The simplest one is a random–random (R–R) algorithm that assumes random selection of K_1 paths and random selection of K_2 content sources between all the content sources that are connected to the users' domain by one of the K_1 selected paths. It is expected that the distance between the solutions selected by algorithm R–R will be high since this algorithm does not select only solutions from the Pareto front but from all the decision space. In order to compare the different algorithms in a complete way, we analyze also the distance of the discovered solutions to the Pareto optimal front.

The next algorithm is Random-MCDM (R-M), which randomly selects K_1 paths in the first phase. The selection of servers uses a variation of reference level multiobjective optimization [43] which resulted efficient for CAN [44]. This algorithm is based on the use of the maximum and the minimum values of each parameter in order to find the appropriate reference levels. Between all the content sources of P_0' (after random selection of paths), the algorithm chooses the solutions that are more distant from the reservation level in order to avoid overload.

The last algorithm, called Domination-MCDM (D-M), selects the non-dominated paths which dominate a higher number of other solutions by applying the formula (9) (minimizing the value of F_i); whereas, the selection of content sources uses the same MCDM algorithm as in the algorithm R-M.

The parameters for selecting paths and servers are ($\mathrm{BW}_{\mathrm{path}}^{-1}$, $\mathrm{IPTD}_{\mathrm{path}}$) and ($\mathrm{BW}_{\mathrm{path}}^{-1}$, $\mathrm{IPTD}_{\mathrm{path}}$, load, $\mathrm{BW}_{\mathrm{access}}^{-1}$), respectively, and the value ε_i is equal to $(\max_j x_{ij} - \min_j x_{ij})/10$, $j \in P_0$.

The testing procedure is the following: in the network scenario presented above, we run each algorithm in order to select the K_1 paths and the $K_2 = 5$ content sources. For each solution, say a found by the algorithm, we calculate its distance to the other $K_2 - 1$ solutions, d^2 (a, others), by applying the formula (13).

$$d^2(a, \text{others}) = \frac{1}{K_2 - 1} \sum_{\substack{b=1 \\ b \neq a}}^{K_2} \sum_{i=1}^{m} \left(\frac{x_{ia} - x_{ib}}{\max\limits_{j, j \in P_0} x_{ij} - \min\limits_{j, j \in P_0} x_{ij}} \right)^2, \quad K_2 = 5;\ m = 4 \tag{13}$$

where $\max_{j, j \in P0} x_{ij}$ and $\min_{j, j \in P0} x_{ij}$ are the maximum and minimum values of the parameter i for all the initial solutions in P_0. Note that the values of $\max_{j, j \in P0} x_{ij}$ and $\min_{j, j \in P0} x_{ij}$ depend only on the network scenario and, because of this, they are the same for all the algorithms and do not have influence in the comparison results. The maximum value of $d^2(a, \text{others})$ is achieved when all the $K_2 - 1$ solutions are very distant from solution a. In that case d^2 (a, others) $= m = 4$.

The distance between solutions d_{sol}^2 for each algorithm is calculated by adding all the five (K_2) distances $d^2(a,\text{others})$ and dividing by 5. This distance represents how distant are the final solutions in each algorithm.

Moreover, we calculate the distance to 0, d_0^2, of the five solutions as indicated in (14).

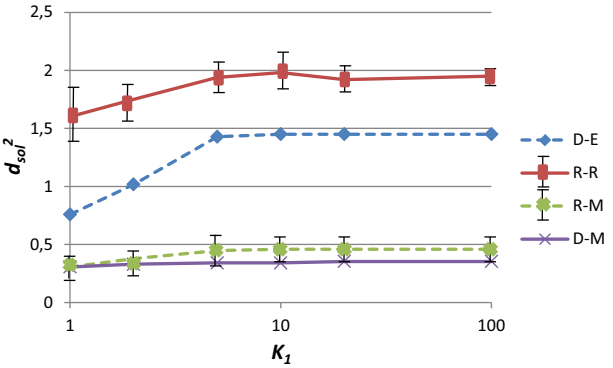

Fig. 6 Results of distance between solutions (d^2_{sol}) for increasing values of K_1 (*D-E* domination-EMO, i.e., the proposed algorithm with $w_1 = w_2 = w_3 = w_4 = 0.5$, *R–R* random–random, *R-M* random-MCDM, *D-M* domination-MCDM)

$$d_0^2 = \frac{1}{K_2} \sum_{a=1}^{K_2} \sum_{i=1}^{m} \left(\frac{x_{ia}}{\max\limits_{j,j \in K} x_{ij}} \right)^2 \quad K_2 = 5; \ m = 4 \tag{14}$$

The value of d_0^2 ($d_0^2 \leq 4$) indicates the mean distance of the solutions to the optimal solution ($0 \in R^m$), so it is an indicator about the effectiveness of the algorithm for finding the best solutions. Let us remark that a good algorithm for CAN should provide high values of d^2_{sol} and low values of d_0^2.

We perform the tests (over the same network scenario) for increasing values of K_1, concretely for the series $K_1 = \{1, 2, 5, 10, 20, 100\}$. Note that, since the average number of servers per domain is 400,000/36,000 ≈11.1, then the initial population for the second phase of the algorithms is, on average, $11.1 \cdot K_1$. The tests were repeated 20 times for the algorithms that select the path randomly (i.e., R–R and R-M) in order to calculate the confidence intervals at the 95 % confidence level that are shown in the figures. Note that these repetitions are to find the average of path selection process (first phase of the algorithm), whereas the second phase is performed only once, so the figure shows results of one unique realization.

Figure 6 shows the values of d^2_{sol}, whereas Fig. 7 presents the d_0^2 results.

The higher values of d^2_{sol} (Fig. 6) in the R–R algorithm indicate that the solutions are widespread, as expected. In fact, this algorithm does not select only solutions of the Pareto front as it can be observed in Fig. 7 (very high values of d_0^2). Therefore, the RR algorithm is not an optimal algorithm in CAN.

After R–R, the highest values of d^2_{sol} are achieved by the D-E algorithm which, besides, is the most effective in finding optimal solutions (see Fig. 7). The algorithms R-M and D-M find solutions that are crowded together, which is inefficient for content source selection purposes even if the solutions are optimal.

A general conclusion is that the proposed D-E algorithm reaches the objectives of CAN infrastructure in a more effective way, obtaining a distribution of the solutions (d^2_{sol}) that is 300 % more spread than MCDM algorithms for the same level (lightly better) of optimization of the final solutions.

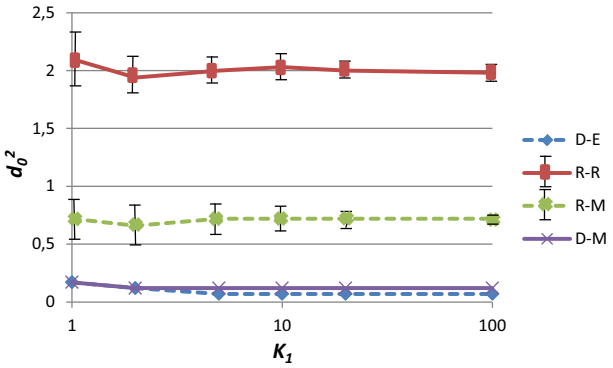

Fig. 7 Results of distance to 0 (d_0^2) for increasing values of K_1 (*D-E* domination-EMO, i.e., the proposed algorithm with $w_1 = w_2 = w_3 = w_4 = 0.5$, *R–R* random–random, *R-M* random-MCDM, *D-M* domination-MCDM)

The confidence intervals (of the results corresponding to the algorithms based on random selection) decrease for increasing values of K_1 (see Fig. 7). This behavior is caused since, for lower values of K_1, the initial population of the second phase P_0' is small and does not represent all the population P_0. Instead, when K_1 is high, the P_0' is a good collection of samples of P_0.

The values of d_0^2 in the D-M algorithm are slightly worse than in the D-E one. This agrees with the conception of the reference level MCDM algorithm that we used. In fact, this algorithm selects the solution by comparing the parameter which is closer to the reservation level (the "worst" parameter) [43,44], while other parameters are not considered. This does not occur in EMO algorithm, which considers all the parameters for content source selection.

We repeated the tests for three other network scenarios (by changing the load of the servers and the size of the network topology, i.e., the network topology is twice larger than in previous tests, so the paths are, in general, longer) and the results were similar. Concretely, the relation of the results between the algorithms were almost identical to the presented ones (the D-E algorithm gives similar values of d_0^2 and three times higher values of d_{sol}^2 than the R-M and D-M algorithms). Therefore, we may conclude that the comparison is not dependent on the network scenario.

In the presented results, we demonstrated the efficiency of the presented algorithm and the better functioning than other algorithms that are used in CAN source selection process. The gaining is achieved specially in obtaining solutions that are spread along the Pareto front, which is necessary in CAN infrastructures. Such a characteristic cannot be achieved by MCDM algorithms since they do not find simultaneously several solutions of the Pareto front.

5 Conclusions

The paper focuses on designing effective decision algorithm for new content delivery architectures called Content-Aware Networks. These networks are able to ensure high

quality in the delivery of multimedia content for critical applications as surveillance and e-health applications. The proposed algorithm aims to select the best content source/server and suitable delivery path based on the information about the content and its requirements, the knowledge of network conditions and the status of available servers. The currently investigated approaches for content source selection are based on multi-criteria decision making, which solves the problem with quite a straightforward solution consisting of the definition of a scalarization function that maps the objective functions onto a single aggregated function.

In this paper, we proposed an algorithm based on Evolutionary Multiobjective Optimization approach. This algorithm has two phases to reflect CAN operations performed on different time scales. The first phase corresponds to the selection of set of suitable delivery paths (based on mid-long time scale resources evaluation), while the second phase selects the potential content sources for particular content request (short time scale). In our algorithm, we introduce novel mechanisms exploiting fitness assignment, variety and elitism to cope specific CAN requirements. *The proposed mechanisms enable the decision-maker to obtain more diversity in the solutions set* than other approaches do: the proposed EMO-based algorithm is more efficient in finding suitable solutions, which are spread along the Pareto front. Such features are very useful in CAN systems and especially in the context of multi-domain and multi-provider environments. The included simulation results demonstrated the profits got by our EMO approach comparing to MCDA approaches usually used in CAN.

Further work will be focused on the application of similar EMO algorithms in the case of multi-source (and multi-path) media streaming, i.e., when a number of sources stream to the end user pieces of content, which are complementary. Such a media delivery is used in collaborative streaming scenarios.

Acknowledgements This work is part of the ODEON project within the European CELTIC+ Programme and DISEDAN project within the European Chist-era Programme. We want to thank the other project partners for their support and contribution to the ideas presented here.

References

1. Ait Chellouche S, Negru D, Arnaud J, Mongay Batalla J (2014) Context-aware multimedia services provisioning in future Internet using ontology and rules. In: The 5th international conference on network of the future (NoF), Paris
2. Bben A, Mongay Batalla J, Wiśniewski P, Xilouris G (2014) A scalable and flexible packet forwarding method for future internet networks. In:IEEE Globecom 2014, Austin
3. Hanne T (1999) On the convergence of multiobjective evolutionary algorithms (Elsevier). Eur J Oper Res 117:553–564
4. Dykes SG, Robbins KA, Jeffery CL (2000) An empirical evaluation of client-side server selection algorithms. In: Proceedings of the IEEE INFOCOM '00, Tel-Aviv
5. Zegura E, Ammar MH, Fei Z, Bhattacharjee S (2000) Application-layer anycasting: a server selection architecture and use in a replicated web service. IEEE/ACM Trans Netw 8(4):455–466

6. Lenders V, May M, Plattner B (2008) Density-based anycast: a robust routing strategy for wireless ad hoc networks. IEEE/ACM Trans Netw 16(4):852–863
7. Koponen T, Chawla M, Chun B-G, Ermolinskiy A, Kim KH, Shenker S, Stoica I (2007) A data-oriented (and beyond) network architecture. In: Proceedings of the ACM SIGCOMM '07, Kyoto
8. Jacobson V, Smetters DK, Thornton James D, Michael P, Nick B, Braynard Rebecca L (2009) Networking named content. In: Proceedings of the ACM CoNEXT 09, pp 1–12
9. Ahlgren Bengt, Aranda Pedro A, Chemouil Prosper, Correia Luis M, Karl Holger, Oueslati Sara, Söllner Michael, Welin Annikki (2011) Connectivity, and cloud: ingredients for the network of the future. IEEE Commun Mag 49(7):62–70
10. Psannis K (2009) Efficient redundant frames encoding algorithm for streaming video over error prone wireless channels. IEICE ELEX J 6(21):1497–1502
11. Sannis KE (2015) HEVC in wireless environments. J Real Time Image Process
12. Kryftis Y, Mavromoustakis CX, Mastorakis G, Pallis E, Mongay Batalla J, Skourletopoulos G (2014) Resource usage prediction for optimal and balanced provision of multimedia services. In: 19th IEEE international workshop on computer aided modeling and design of communication links and networks (CAMAD), Athens
13. Garcia A, Rodriguez L, Hackbarth K (2011) Cost models for QoS-differentiated interconnecting and wholesale access services in future generation networks. Springer Telecommun Syst. doi:10.1007/s11235-011-9431-6
14. Jung J, Chang Y, Yu Z (2011) Advances on intelligent network management. Springer Telecommun Syst. doi:10.1007/s11235-011-9507-3
15. Hoffert J, Gokhale A, Schmidt DC (2011) Timely autonomic adaptation of publish/subscribe middleware in dynamic environments. Int J Adapt Resilient Auton Syst 2(4):1–24
16. Su A, Choffnes DR, Kuzmanovic A, Bustamante FE (2009) Drafting behind Akamai: inferring network conditions based on CDN redirections. In: IEEE/ACM transactions on networking, vol 17(6)
17. Beben A, Mongay Batalla J, Chai W, Sliwinski J (2013) Multi-criteria decision algorithms for efficient content delivery in content networks. Ann Telecommun Ann Telecommun 68(3):153–165
18. Li D, Das S, Pahwa A (2009) Two-phase multi-objective evolutionary approaches for optimal generation scheduling with environmental considerations. In: North American power symposium (NAPS)
19. Dubois-Lacoste J, López-Ibáñez M, Stützle T (2013) Combining two search paradigms for multi-objective optimization: two-phase and Pareto local search. In Talbi E-G (ed) Hybrid metaheuristics. Studies in computational intelligence, vol 434. Springer, New York
20. Israel S, Moshaiov A (2012) Bootstrapping aggregate fitness selection with evolutionary multi-objective optimization. In: PPSN'12 proceedings of the 12th international conference on parallel problem solving from nature, vol Part II, pp 52–61
21. Lakshminarasimman N, Baskar S, Alphones A, Willjuice Iruthayarajan M (2011) Evolutionary multiobjective optimization of cellular base station locations using modified NSGA-II. In: Springer (eds) Wireless networks, vol 17, pp 597–609
22. Psannis K, Ishibashi Y (2006) Impact of video coding on delay and jitter in 3G wireless video multicast services. EURASIP J Wirel Commun Netw 2006(24616):1–7
23. Kryftis Y, Mavromoustakis CX, Mastorakis G, Pallis E, Mongay Batalla J, Rodrigues JPC, Dobre C, Kormentzas G (2015) Resource usage prediction algorithms for optimal selection of multimedia content delivery methods. In: IEEE international conference on communications ICC, London
24. Wiegand S, Igel C, Handmann U (2004) Evolutionary multi-objective optimization of neural networks for face detection. Int J Comput Intell Appl 04(237):2004. doi:10.1142/S1469026804001288
25. Padhye N, Zuo L, Mohan CK, Varshney PK (2009) Dynamic and evolutionary multi-objective optimization for sensor selection in sensor networks for target tracking. In: Proceedings of the international joint conference on computational intelligence, Funchal, Madeira, 5–7 October 2009, pp 160–167. INSTICC Press (ISBN 978-989-674-014-6)
26. Pullan W (2002) Optimising multiple aspects of network survivability. In: Proceedings of the 2002 congress on evolutionary computation, 2002, CEC '02, vol 1, pp 115–120, Honolulu
27. Horoba C, Neumann F (2008) Benefits and drawbacks for the use of epsilon-dominance in evolutionary multi-objective optimization. In: Proceedings of the genetic and evolutionary computation conference (GECCO 2008)
28. Coello Coello CA (2006) Evolutionary multi-objective optimization: a historical view of the field. In: IEEE computational intelligence magazine, vol 1(1), pp 28–36

29. Knowles JD, Corne DW (2000) Approximating the nondominated front using the Pareto archived evolution strategy. Evolut Comput 8(2):149–172
30. Okei CK, Goldberg DE, Chang S (1991) Tournament selection, Niching, and the preservation of diversity. In: Technical report 91011, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana
31. Erickson M, Mayer A, Horn J (2001) The niched Pareto genetic algorithm 2 applied to the design of groundwater remediation systems. Lecture notes in computer science 1993:681–695
32. Zitzler E, Laumanns M, Thiele L (2002) SPEA2: improving the strength Pareto evolutionary algorithm. In: Giannakoglou K et al (eds) EUROGEN 2001, evolutionary methods for design, optimization and control with applications to industrial problems, Athens, pp 95–100
33. Zitzler E, Laummanns M, Thiele L (2002) SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization. In: Evolutionary methods for design, optimisation and control, Barcelona, pp 19–26
34. Lara A, Sanchez G, Coello Coello CA, Schutze O (2010) HCS: a new local search strategy for memetic multiobjective evolutionary algorithms. IEEE Trans Evolut Comput 14(1):112–132
35. Wagner M, Friedrich T (2013) Efficient parent selection for approximation-guided evolutionary multi-objective optimization. In: Proceedings of the IEEE conference on evolutionary computation
36. University of Oregon (2016) Route views archive project David Meyer. http://archive.routeviews.org/. Accessed 1 April 2016
37. RIPE Network Coordination Centre (2016) Routing information service. http://www.ripe.net/data-tools/stats/ris/ris-peering-policy. Accessed 1 April 2016
38. Akamai Technologies Inc. (2016) Facts & figures. http://www.akamai.com/html/about/facts_figures.html. Accessed 1 April 2016
39. Yu H, Zheng D, Zhao B, Zheng W (2006) Understanding user behavior in large-scale video-on-demand systems. In: Proceeding of the EuroSys
40. Agarwal S, Nucci A, Bhattacharyya S (2005) Measuring the shared fate of IGP engineering and interdomain traffic. In: Proceedings of the 13th international conference on network protocols (ICNP)
41. Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans Evolut Comput 3(4):257–271
42. Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms: empirical results. J Evolut Comput 8(2):173–195
43. Wierzbicki A (2016) The use of reference objectives in multiobjective optimization. In: Lecture notes in economics and mathematical systems, vol 177. Springer, New York, pp 468–486
44. Mongay Batalla J, Bben A, Chen Y (2015) Optimized decision algorithm for information centric networks. Springer Telecommun Syst J. doi:10.1007/s11235-015-9998-4